

**UTILITY
PATENT APPLICATION
TRANSMITTAL**

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No.

P98,3

First Named Inventor or Application Identifier

Karlheinz Dorn et al,

Express Mail Label No: # EL121676562US

ADDRESS TO: Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

1. ☒ Specification [Total Pages 27]
2. ☒ Drawing(s) (35USC 113) [Total Pages 10]
3. ☒ Declaration and Power of Attorney [Total Pages 3]

a. ☐ Newly executed(original copy)

b. ☐ Copy from prior application (37CFR 1.63(d))
(for continuation/divisional with Box 14 completed)

- i. ☐ **[Note Box 4 Below]**
DELETION OF INVENTOR(S)
Signed statement attached deleting
Inventor(s) named in the prior application,
see 37 CFR 1.63(d)(2) and 1.33(b).

4. ☐ Incorporation By Reference (usable if Box 3b is checked)
The entire disclosure of the prior application, from which a
copy of the oath or declaration is supplied under Box 3b,
is considered as being part of the disclosure of the
accompanying application and is hereby incorporated by
reference therein.

ACCOMPANYING APPLICATION ELEMENTS

5. ☐ Assignment Papers (cover sheet & documentation)
6. ☒ Letter under 37 CFR 1.41(c).
7. ☐ English Translation Document (if applicable)
8. ☐ Information Disclosure Statement (IDS)/PTO-1449 ☐ Copies of IDS Citations
9. ☐ Preliminary Amendment
10. ☒ Return Receipt Postcard (MPEP 503)
(Should be specifically itemized)
11. ☐ Small Entity Statement filed in prior application, Status still proper and desired
12. ☐ Certified Copy of Priority Document(s).
13. ☐ Other: _____

14. If a CONTINUING APPLICATION, check appropriate box and supply the requisite information:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) ☐ of prior application No: _____/_____

CLAIMS AS FILED

	(1) FOR	(2) NUMBER FILED	(3) NUMBER EXTRA	(4) RATE	(5) BASIC FEE \$760.00
TOTAL CLAIMS	20	5			
INDEPENDENT CLAIMS	3	1			
	ANY MULTIPLE DEPENDENT CLAIMS? (YES (X) NO				
				TOTAL FILING FEE ->	\$760.00

☒ The Commissioner is hereby authorized to charge any additional fees which may be required in connection with this application, or credit any overpayment to ACCOUNT NO. 08-2290. A duplicate copy of this sheet is enclosed.

☒ A check in the amount of \$ 760.00 to cover the filing fee is enclosed.

15. CORRESPONDENCE ADDRESS

HILL & SIMPSON
A Professional Corporation
233 South Wacker Drive - 85th Floor Sears Tower
Chicago, Illinois 60606
Telephone (312) 876-0200 - Fax (312) 876-0898

SIGNATURE: Daniel M. Meyer
491/1044:1190

DATE: December 18, 1998

U-11

CERTIFICATE OF MAILING

"Express Mail" Mailing Label Number **EL 121676562 US**



Date of Deposit: December 18, 1998

I hereby certify that this correspondence is being deposited with the United States Postal "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to:

BOX PATENT APPLICATION
Assistant Commissioner for Patents
Washington, D.C. 20231

Case Number: P98,3059

Inventors: Karlheinz Dorn et al.

Patent Application entitled:

"INTERPRETIVE NETWORK DAEMON IMPLEMENTED BY GENERIC MAIN
OBJECT"

Signature of person mailing documents and fee

A handwritten signature in black ink, consisting of a stylized 'A' followed by a series of loops and a horizontal line at the bottom.

HILL & SIMPSON
A PROFESSIONAL CORPORATION
ATTORNEYS AND COUNSELORS AT LAW
CHICAGO, ILLINOIS 60606

JOHN D. SIMPSON *
LEWIS T. STEADMAN
JAMES A. MOEHLING
DENNIS A. GROSS
ROBERT M. BARRETT
STEVEN H. NOLL
KEVIN W. GUYNN
SCOTT W. PETERSEN
ROBERT M. WARD
BRETT A. VALIQUET
GEORGE C. SUMMERFIELD
LEWIS T. STEADMAN, JR.
EDWARD A. LEHMAN
DAVID R. METZGER
TODD S. PARKHURST
JOHN R. NYWEIDE
JAMES D. HOBART
MELVIN A. ROBINSON
JOHN R. GARRETT
C. GRANT MCCORKHILL

PAULA J. KELLY
JOHN W. CORNELL
—
ROBERT J. DEPKE
PATRICIA A. KANE
JOSEPH P. REAGEN
MICHAEL R. HULL
MICHAEL S. LEONARD
WILLIAM E. VAUGHAN
—
JAMES VAN SANTEN
J. ARTHUR GROSS
MARVIN MOODY
—
DOLORES K. HANNA
SPECIAL TRADEMARK COUNSEL

CHICAGO OFFICE
85TH FLOOR SEARS TOWER
CHICAGO, ILLINOIS 60606
TELEPHONE (312) 876-0200
FACSIMILE (312) 876-0898
INTERNET: counsel@hillfirm.com

WASHINGTON OFFICE
SUITE 1004-BLDG. I
2001 JEFFERSON DAVIS HIGHWAY
CRYSTAL CITY
ARLINGTON, VIRGINIA 22202
TELEPHONE (703) 415-1515

* MUNICH OFFICE
FRANZ-JOSEPH STRASSE 38
D-80801 MUNICH, GERMANY
49-89-3840720

December 18, 1998

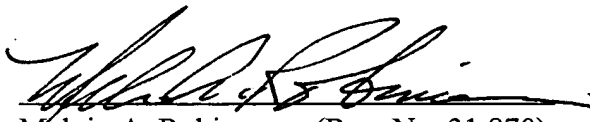
Assistant Commissioner for Patents
Washington, D.C. 20231

Re: U.S. patent application for DORN et al., "INTERPRETIVE NETWORK DAEMON
IMPLEMENTED BY GENERIC MAIN OBJECT", Our File No. P98,3059

S I R:

Under the provisions of 37 CFR 1.41 (c), I am filing the attached patent application on behalf of Karlheinz Dorn et al. an unexecuted Declaration being submitted herewith, and request that the application papers be assigned a serial number and filing date pursuant to the provisions of 1.53(b) and 1.53(d).

Respectfully submitted,


Melvin A. Robinson (Reg. No. 31,870)

1044/1072
Enclosures



23612/60
Old-S-N-5250JF

/* Compilation unit -----

Component : CSA -

Name : CsaGenericChainBas.h

Author : (D. Quehl, BHEP1, +49 9131 84 2430); Siemens AG B Med CSA

Language : C/C++

Creation Date : 15-JUN-1996

Test State :

Description : Definition of Generic Main functionality.

Requirement Key : cs_os_process_mqr

Copyright (C) Siemens AG 1995 All Rights Reserved

/* END */

#ifndef CSAGENERICHAINBAS_H
#define CSAGENERICHAINBAS_H

/* program comment (exect: "4(1) CsaGenericChainBas.h 1.2 (10 Jul 1996) : Definition Generic Main Base Class")

#include <ace/OS.h>

#include <ace/Task.h>

#include <ace/Synch.h>

#include <ace/Synch_T.h>

#include <CsaCommon/CsaDefs.h>

#include <os/CsaGack/MessageId.h>

/* Data type -----

Name : CSA_STATUS - macro for Msc component

Description : Macro for error Handling

-- uses status as input
-- hides setting of line and file

/*

#ifndef CSA_ERR_DISABLE

#include <err/CsaStatus.h>

#define CSA_STATUS_gmain(status) | csaStatus->set(status, FILE, LINE, 1)

#else

#define CSA_STATUS_gmain(status) ;

#endif // #ifndef CSA_ERR_DISABLE

#define CSA_STATUS_gmain(status) CSA_STATUS_gmain(CSA_OSC_MSC_Hstatus)

/* END Data type */

/* Data type -----

Name : CSA_TRACE_IN - Macro redefinition.

Description : Macro redefinition that disables method trace.

#ifndef CSA_TRACE_DISABLE

#include <err/CsaTrace.h>

#else

#define CSA_TRACE_IN(a)

#endif // #ifndef CSA_TRACE_DISABLE

/* END Data type */

/* Class -----

Name : CsaGenericChainBas

Description : CsaGenericChainBas provides the generic main program
functionality. CsaGenericChainBas is derived from
ACE_Task<ACE_MT_SYNCH> to let the dispatch loop
run in either, the main (caller) thread or a
separate thread (task's svc method).

class CsaGenericChainBas : public ACE_Task<ACE_MT_SYNCH> {

public:

// Constructor/destructor

CsaGenericChainBas(void);

CsaGenericChainBas(int argc, char *argv[]);

~CsaGenericChainBas(void) {

delete this; }

};

// Store commandline arguments if instantiated using the default

void setArgs (int argc, char *argv[]) {

argc_ = argc;

argv_ = argv;

};

// handle SIGINT

virtual int handle_signal (int signum);

// Perform initialization steps for CsaGenericChainBas

BEST AVAILABLE COPY

```

virtual int open (void *p = 0);

// Invoke dispatch loop [a]synchronously.
virtual int dispatchloop(bool async = false);

// synchronize on dispatcher termination
virtual void sync(void);

// break the loop
virtual int endloop(void);

// hook methods that can be overloaded to insert functionality before
// the loop starts and immediately after the loop finished.
virtual int loophook1(void) { return 0; }
virtual int loophook2(void) { return 0; }

// hook methods that can be overloaded to insert functionality prior
// to daemon open and after daemon close.
virtual int openhook(void) { return 0; }
virtual int closehook(void) { return 0; }

private:

// The implementation of the main dispatch loop.
virtual int svc (void);

// The following methods are inherited from ACE_Task which defines
// them as pure virtual methods.
// They are not needed here and implemented returning instant error.
// Subclasses of CsaGenericChainBas may overload them.
virtual int put (ACE_Message_Block *, ACE_Time_Value * = 0) {
    return -1;
}
virtual int close (unsigned long flags = 0) {
    return 0;
}
virtual int handle_input (ACE_HANDLE fd = ACE_INVALID_HANDLE) {
    return -1;
}

// The service configuration daemon
Service_Config daemon_;

// Copy of the command line arguments
int argc_;
char **argv_;

// Indicator for initialization state
bool initialized_;

// API lock that prevents loop from being called multiply
// and synchronizes the termination of the thread that runs
ACE_Thread_Mutex lock_;

// provide ACE_Task a thread manager

```

```

ACE_Thread_Manager *thmgr_;
};

/*| END Class */

#endif //!ifndef CsaGenericChainBAS_H

/*| Change header -----
Date       : 26. Jun 1996
Version    : -
Charm      : -
Author     : D. Quehl (Qu); Siemens AG B Med CSA
Description : Initial release
-----*/

/*| Change header -----
Date       : 07. Jul 1996
Version    : 1.1
Charm      : 2041
Author     : D. Quehl (Qu); Siemens AG B Med CSA
Description : Changes in respect to code review component
             *osc - generic main - 03.jul.96 *
-----*/

```

```

/*| Compilation unit -----

```

```

Component : CSA -

```

```

Name : CsaGenericChainBas.cpp

```

```

Author : D. Quehl, BNEP11, +49 9131 84 2430; Siemens AG B Med CSA

```

```

Language : C/C++

```

```

Creation Date : 19-JUN-1996

```

```

Test State :

```

```

Description : Implementation of the Generic Main functionality.

```

```

Requirement Key : os_os_process_mgr, os_os_test_testability

```

```

Copyright (C) Siemens AG 1995 All Rights Reserved

```

```

/*| END */

```

```

#pragma comment ( exesit, "44) CsaGenericChainBas.cpp 1.3 (10 Jul 1996) : Implementation Generic Main Base class" )
#include <os/CsaGenericChainBas.h>

```

```

/*| Method -----

```

```

Name : CsaGenericChainBas::CsaGenericChainBas

```

```

Description : Default constructor for CsaGenericChainBas.
              Prior to starting the dispatch loop the
              command line arguments must be stored using
              the setArgs() method.
              The constructor calls the open() method.

```

```

Return : none

```

```

CsaGenericChainBas::CsaGenericChainBas(
    void // takes no arguments
)

```

```

/*| END Method */

```

```

: argc_(0), argv_(0), initialized_(false)

```

```

CSA_TRACE_IN ((CSA_OSC, "int CsaGenericChainBas::CsaGenericChainBas()");
              (this->open());

```

```

/*| Method -----

```

```

Name : CsaGenericChainBas::CsaGenericChainBas

```

```

Description : Constructor for CsaGenericChainBas.
              Stores the command line arguments and calls the open()
              method.

```

```

Return : none

```

```

CsaGenericChainBas::CsaGenericChainBas(
    int argc, // argument count
    char *argv[] // argument vector
)

```

```

/*| END Method */

```

```

: initialized_(false)

```

```

CSA_TRACE_IN ((CSA_OSC, "int CsaGenericChainBas::CsaGenericChainBas()");

```

```

              (this->setArgs(argc,argv));
              (this->open());

```

```

/*| Method -----

```

```

Name : CsaGenericChainBas::open

```

```

Description : Perform initialization steps for CsaGenericChainBas.
              - Create our own thread manager

```

```

Return : 0 success
        -1 error

```

```

int CsaGenericChainBas::open (
    void *p // pointer for arbitrary use
)

```

```

/*| END Method */

```

```

CSA_TRACE_IN ((CSA_OSC, "int CsaGenericChainBas::open()");

```

```

if ((this->thmgr_ = new ACE_Thread_Manager) == 0) {
    CSA_STATUS_GMAIN(NOWEN_NEW);
    return -1;
}

```

```

(this->thmgr(thmgr_)) // pass thread manager to task
return 0;

```

```

/*| Method -----

```

```

Name      : CsaGenericChainBas::handle_signal

Description : The handle_signal() method is called on an incoming
             SIGINT.
             handle_signal() calls endloop() to terminate dispatching.

Return     : 0      success
            -1      error

-----*/
int CsaGenericChainBas::handle_signal (
    int signum // signal number that caused invocation
)
{
    /* END Method */

    A_TRACE_IN ((CSA_OSC, "int CsaGenericChainBas::handle_signal"));
    if (this->thmgr != 0)
        return this->endloop();
    CSA_STATUS_OPAIN(NOWEM_NEW);
    return -1;
}

/* Method -----
Name      : CsaGenericChainBas::svc

Description : The svc() method implementation of the main dispatch loop.
             After starting the ACE_Service_Config daemon with the
             stored command line arguments the first hook method is
             called and - if this hook returns success - the dispatch
             loop invoked.
             Immediately after returning from the dispatch loop the
             second hook method is called to perform subclass specific
             cleanup actions while the ACE_Service_Config is still up.

Return     : 0      success
            -1      error

-----*/
int CsaGenericChainBas::svc (
    void
) // takes no arguments
{
    /* END Method */

    CSA_TRACE_IN ((CSA_OSC, "int CsaGenericChainBas::svc"));
    // return error if thread manager creation failed in open()
    if (this->thmgr == 0) {
        CSA_STATUS_OPAIN(NOWEM_NEW);
    }
}

```

```

        return -1;
    }

    // Lock the access to the dispatch loop
    ACE_TSS_Guard<ACE_Thread_Mutex> mon (this->lock);

    // make us owner of the reactor
    ACE_Service_Config::reactor()->owner(ACE_Thread::self());

    // invoke the starthook
    if (this->openhook() == -1)
        return -1;

    // Start the daemon
    ACE_SEM_TRY {
        if (this->daemon_open (this->argc, this->argv) == -1) {
            CSA_STATUS_OPAIN(NO_REACTOR);
            return -1;
        }
    }

    ACE_SEM_EXCEPT(EXCEPT_EXECUTE_HANDLER) {
        cout << "Caught structured exception while opening Service_Config daemon" << endl;
    }

    // Register this to the reactor to receive SIGINT
    if (ACE_Service_Config::reactor()->register_handler (SIGINT, this) == -1) {
        CSA_STATUS_OPAIN(REGISTER_HANDLER);
        (void)this->daemon_close();
        return -1;
    }

    // Call the first hook; fail through if hook returns an error
    if (this->lookhook1() >= 0) {
        initialized = true;
        ACE_SEM_TRY {
            ACE_Service_Config::run_event_loop ();
        }
        ACE_SEM_EXCEPT(EXCEPT_EXECUTE_HANDLER) {
            cout << "Caught structured exception while running Service_Config event loop" << endl;
        }
        (void)this->lookhook2();
    }

    int status;

    // Close down the services of daemon
    ACE_SEM_TRY {
        status = this->daemon_close_svc();
    }
    ACE_SEM_EXCEPT(EXCEPT_EXECUTE_HANDLER) {
        cout << "Caught structured exception while closing down Service_Config daemon" << endl;
    }

    // invoke the close hook
    (void)this->closehook();
}

```

```

// Close down the memory of daemon
status = this->daemon_close_mem();

initialized_ = false;

return status;
}

```

```

/*| Method -----
Name      : CsaGenericChainBas::dispatchloop

```

```

Description : Invoke dispatch loop either synchronously (in the context
of the caller thread) or asynchronously (using
ACE_Task::activate() to create a separate thread).

```

```

return      : 0          success
              -1         error

```

```

int CsaGenericChainBas::dispatchloop(
    bool async // flag for asynchronous invocation
)

```

```

/*| END Method */

```

```

// return error if thread manager creation failed in open()
if (this->thmgr_ != 0) {
    CSA_STATUS_CHAIN(NOMEM_NEM);
    return -1;
}

```

```

// check whether initialization was already set
if (!initialized_) {

```

```

// for synchronous operation just invoke svc method in the context
// of the caller's thread.
if (!async)
    return this->svc();

```

```

// for asynchronous operation invoke svc method in the context
// of a thread created by activate().
else

```

```

    return this->activate();
}

```

```

// Return error if invoked twice.
return -1;

```

```

/*| Method -----

```

```

Name      : CsaGenericChainBas::endloop

```

```

Description : Stop dispatching.
Return      : 0          success
              -1         error

```

```

int CsaGenericChainBas::endloop(
    void // takes no arguments
)

```

```

/*| END Method */

```

```

ACE_Service_Config::end_event_loop();
return 0;
}

```

```

/*| Method -----
Name      : CsaGenericChainBas::sync

```

```

Description : Synchronize on dispatcher thread finished
Return      : none

```

```

void CsaGenericChainBas::sync(
    void // takes no arguments
)

```

```

/*| END Method */

```

```

if (this->thmgr_ != 0)
    this->thmgr_>wait();
ACE_Thread::exit(0);
}

```

```

/*| Change header -----

```

```

Date       : 26. Jun 1996
Version    : -
Charm     : -
Author    : D. Quehl (Qu); Siemens AG 8 Med CSA
Description : Initial release

```

```

/*| Change header -----

```

```

Date       : 07. Jul 1996

```


Version : 1.1
Charm : 2061
Author : D. Quehl (qu) ; Siemens AG B Med CSA

Description : Changes in respect to code review component
"osc - generic main - 03.jul.96"

*/

```
/*| Compilation unit -----
Component      : CSA -
Name           : CsGenericMain.h
Author         : (D. Quehl, BNEP1, +49 9131 84 2430); Siemens AG B Med CSA
Language       : C/C++
Creation Date  : 19.-JUN-1996
Test State    :
Description    : Add on to Generic Main functionality.
                  Definition of library startup/shutdown functionality.
Requirement Key : os_process_mgr

Copyright (C) Siemens AG 1995 All Rights Reserved

/*| END */
#ifndef CSAGENERICMAIN_H
#define CSAGENERICMAIN_H

#pragma comment ( exestr, "4(1) CsGenericMain.h 1.4 (10 Jul 1996) test: Definition Generic Main" )
#include <os/CsGenericMainBas.h>

// Header files of Singletons
#include <os/CsSesam.h>
#include <os/CsAwsComm.h>
#include <os/CsOsSystem.h>
#include <os/CsMrgCndf.h>

/*| Class -----
Name      : CsGenericMain

Description : CsGenericMain is a friend class of a couple of
              basic libraries all designed using the singleton
              pattern with a private constructor/destructor and
              destroy() method. These singletons will be instantiated
              in the open method of CsGenericMain and destroyed
              (in reverse order) in the close() method.

class CsGenericMain : public CsGenericMainBas {
public:
    // Constructor
    CsGenericMain(int argc, char *argv[])

```

```
    : CsGenericMainBas(argc,argv) {}

    // The openhook() method is called by the svc() method and instantiates
    // all singletons the generic main provides for the component DLLs.
    virtual int openhook (void);

    // The closehook() method is called by the svc() method to close up
    // all singletons the generic main provides for the components
    // (reverse order).
    virtual int closehook (void);

    // Instances of of singletons
    CsAwsComm *Comm_;
    CsSesam *sesam_;
    CsOsSystem *os_;
    CsMrgCndf *rti_;
};

/*| END Class */

#endif //ifndef CSAGENERICMAIN_H

/*| Change header -----
Date      : 26. Jun 1996
Version   : -
Charm     : -
Author    : D. Quehl (Qu); Siemens AG B Med CSA
Description : Initial release

```

```

/*| Compilation unit -----
Component      : CSA
Name           : CsaGenericChain.cpp
Author          : (D. Quethl, BREFFI, *49 9131 84 2430); Siemens AG B Med CSA
Language        : C/C++
Creation Date   : 19-JUN-1996

Test State      :

Description      : Add on to Generic Main functionality.
Implementation of library startup/run/dow functionality.

Requirement Key : OS_CS_PROCESSING, OS_CS_TEST_TESTABILITY

Copyright (C) Siemens AG 1995 All Rights Reserved

-----*/
/*| END Method */

/*| Private comment ( exest; *49 9131 84 2430) test: Implementation Generic Main */
#include <os/CsaGenericChain.h>

/*| Method -----
Name           : CsaGenericChain::openhook
Description      : The openhook() method is called by the svc() method and
                  instantiates all singletons the generic main provides
                  for the component DLLs.
Return          : 0          success
                  -1         error
-----*/
int CsaGenericChain::openhook (
    void          // takes no arguments
)
{
    /*| END Method */

    CSA_TRACE_IN ((CSA_OS, "int CsaGenericChain::openhook"));
    this->sesam_ = CsaSesam::sesam();
    if (this->sesam_ == 0) {
        CSA_STATUS_GMAIN(NOMEM_NEW);
    }
    this->commu_ = CsaScComm::commu();
    if (this->commu_ == 0) {

```

```

        CSA_STATUS_GMAIN(NOMEM_NEW);
    }
    this->os_ = CsaOsSystem::csaOsSystem();
    if (this->os_ == 0) {
        CSA_STATUS_GMAIN(NOMEM_NEW);
    }
    this->rti_ = CsaPrtCondIf::csaPrtCondIf();
    if (this->rti_ == 0) {
        CSA_STATUS_GMAIN(NOMEM_NEW);
    }
    return 0;
}

/*| Method -----
Name           : CsaGenericChain::closehook
Description      : The closehook() method is called by the svc() method and
                  destroys all singletons the generic main provided for the
                  components (reverse order).
Return          : 0          success
                  -1         error
-----*/
int CsaGenericChain::closehook (
    void          // takes no arguments
)
{
    /*| END Method */

    // close down singletons
    // cout << "CsaGenericChain::closehook>> closing down libraries" << endl;
    // cout << "CsaPrtCondIf" << endl;
    // this->rti_>destroy();
    // cout << "CsaOsSystem" << endl;
    // this->os_>destroy();
    // cout << "CsaSc" << endl;
    // this->commu_>destroy();
    // cout << "CsaSesam" << endl;
    // this->sesam_>destroy();

    // cout << "CsaGenericChain::closehook>> closing down libraries finished" << endl;
    return 0;
}

/*| Change header -----

```

Date : 26. Jun 1996
Version : -
Cham : -
Author : D. Quehl (Qu); Siemens AG B Med CSA
Description : Initial release

/* Change header
Date : 07. Jul 1996
Version : 1.1
Cham : 2041
Author : D. Quehl (Qu); Siemens AG B Med CSA
Description : Changes in respect to code review component
*osc - generic main - 03.jul.96 *

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.